UNIVERSITÉ DE MONTRÉAL

CENTRE DE RECHERCHE SUR LES TRANSPORTS

CASE POSTALE 6128, SUCCURSALE A, MONTRÉAL, QUÉBEC, H3C 3J7

AN ACTIVE CONSTRAINT LINEARIZED JACOBI NEWTON ALGORITHM FOR MONOTONE VARIATIONAL INEQUALITY WITH APPLICATION TO THE GSPEP

by

Omar Drissi-Kaïtouni

Cette étude a été publiée grâce à une subvention du fonds F.C.A.R. pour l'aide et le soutien à la recherche.

Université de Montréal

Centre de recherche sur les transports - Publication #547

Département d'Informatique et
de Recherche opérationnelle - Publication #634

June 1987

Dépôt légal - 4e trimestre 1987

ABSTRACT

In this paper, we consider the General Spatial Price Equilibrium Problem (GSPEP), and we show that by using a simple modification of the notation, the GSPEP can be seen as a Network Equilibrium Problem (NET). In the most general case (asymmetric costs), these two problems may be written as variational inequality problems subject to linear equality constraints and nonnegativity constraints (and, sometimes, capacity constraints).

To solve this kind of problem, we propose a linearized-Jacobi method in which every subproblem is solved by a Newton method. The subproblems are shown to be equivalent to quadratic programs with linear equality constraints and nonnegativity constraints (there may also be capacity constraints). We use a Newton method with an active constraint set strategy to solve the subproblems.

The global algorithm that we call LJN (for Linearized Jacobi Newton) has been implemented on the CYBER 835 of the Université de Montréal via a Fortran code, and the numerical results, obtained on several test networks reflect the efficiency of that algorithm for monotone variational inequality (and convex mathematical programs) with linear equality and nonnegativity (and capacity) constraints.

Key words: variational inequality, linearized Jacobi, Newton, active constraints, General Spatial Equilibrium.

RÉSUMÉ

Dans cet article, nous considérons le Problème d'équilibre spatial sur réseau général (PESG), et en utilisant une simple modification de la notation, nous démontrons que le PESG peut être perçu comme un Problème d'équilibre de réseau (PER) à une commodité. Dans le cas le plus général (coûts assymétriques), ces deux problèmes peuvent être formulés comme des inégalités variationnelles sujet à des contraintes d'égalité linéaires et des contraintes de non-négativité (et, parfois, à des contraintes de capacité).

Afin de résoudre ce type de problème, nous proposons une méthode de type Jacobi-linéarisé où chaque sous-problème est résolu par la méthode de Newton. On démontre que les sous-problèmes sont équivalents à des programmes quadratiques avec contraintes d'égalité linéaires et des contraintes de non-négativité (il peut aussi y avoir des contraintes de capacité). Nous utilisons une méthode de Newton avec une stratégie de contraintes actives pour résoudre les sous-problèmes.

L'algorithme global que nous dénotons JLN (pour Jacobi-linéarisé-Newton) a été implanté sur le CYBER 835 de l'Université de Montréal via un code Fortran, et les résultats numériques, obtenus sur plusieurs réseaux tests, reflètent l'efficacité de l'algorithme pour les problèmes d'inégalités variationnelles monotones (et programmes mathématiques convexes) avec contraintes d'égalités linéaires et de non-négativité (et de capacité).

Mots-clés : inégalités variationnelles, Jacobi-linéarisé, Newton, contraintes actives, équilibre spatial général.

1. INTRODUCTION

In this paper, we deal with the Spatial Price Equilibrium Problem on a General Network, also known as the General Spatial Price Equilibrium Problem (GSPEP).

As an illustration of this problem, in figure 1-1 one finds the Spatial Price Equilibrium Problem (SPEP) (on a bipartite network) which is a particular case of the GSPEP. Due to its similarity with the well-known Hitchcock (1941) Transportation Problem, the SPEP is also known as the Generalized Transportation Problem. In the SPEP, some goods are produced at origins i and consumed at destinations j. One then seeks to determine, for every pair O/D (i,j), the equilibrium between the production cost (defined by a monotone increasing supply cost function S_i) plus the transportation cost (using the monotone increasing transportation cost C_{ij}), and the market price (characterized by a monotone decreasing, demand function I_j). We refer to (Zubieta, 1987) for solving this kind of problem.

In this paper, we consider the more general case (figure 1-2), where origins and destinations are on a general network. In this case, an O/D pair (i,j) is no longer connected by a link, but by many paths. The problem remains the same as in the SPEP but the transportation cost between any pair O/D (i,j) is defined by using the transportation costs on every link of the given path between (i,j).

We would like to mention that in the present paper, we consider the general case of the GSPEP where, on every link a, of the general network, the transportation cost C_a depends on the flow v_a on this link, but may also depend on flows v_b of other links b of the network.

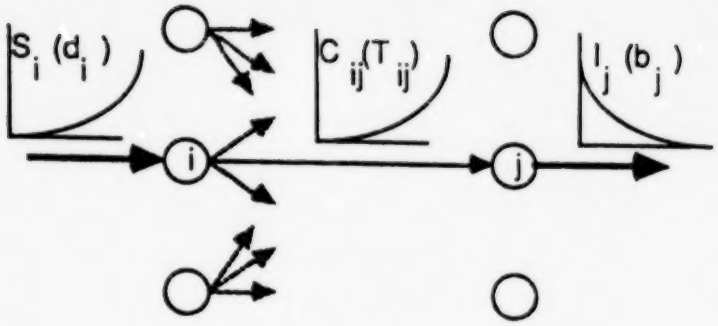

Figure 1-1: Spatial Price Equilibrium Problem

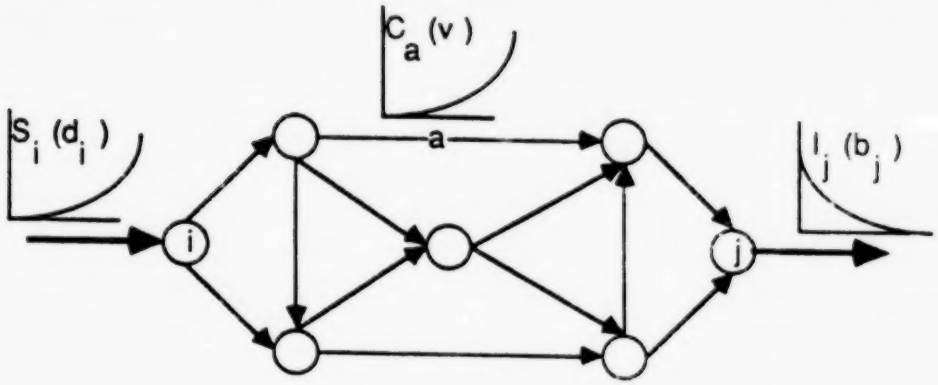

Figure 1-2: General Spatial Price Equilibrium Problem

The remainder of the paper is as follows: in the next section, we recall the mathematical formulation of the problem; in section 3, we deal with a Jacobi-Newton type algorithm to solve a general variational inequality with linear equality and nonnegativity constraints. In section 4, we consider implementation details of the previous algorithm to the GSPE problem. Numerical results are presented in section 5, followed by a conclusion.

2. FORMULATION OF THE GSPEP

In this section, we recall the formulation of the GSPEP in terms of path flow variables as well as in terms of arc flow variables. But in the remainder of this paper we deal only with the arc flow formulation. We refer to (Drissi, 1987; Drissi and Florian, 1987) for the path flow formulation; yet, those papers consider the symmetrical case in which the transportation costs on every arc a depend only on the flow v_a on the same arc a.

Let w be the set of origin-destination pairs (i,j); some goods are produced at origins i and consumed at destinations j. The production cost p_i, at origin i, is defined by a monotone increasing supply function $p_i = S_i(d_i)$ where d_i is the quantity produced at origin i, and the market price m_j is defined by a monotone decreasing demand function $m_j = I_j(b_j)$, where b_j is the quantity of goods demanded at destination j. Let C_{ij} the transportation cost between origin i and destination j. The monotonicity conditions on functions S, C, and I are here to ensure existence of an equilibrium (Smith, 1979; Dafermos, 1980).

On a bipartite network, when the transportation cost (per unit) C_{ij} is independant of the quantity T_{ij} transported between any pair (i,j), Samuelson (1952) stated the equilibrium conditions as:

$$\begin{aligned} S_i(d_i) + C_{ij} - I_j(b_j) &= 0 \quad \text{if} \quad T_{ij} > 0 \quad \forall (i,j) \\ &> 0 \quad \text{if} \quad T_{ij} = 0 \quad \forall (i,j) \end{aligned} \qquad (2\text{-}1)$$

where d_i, T_{ij}, and b_j are subjected to conservation flow conditions:

$$d_i = \sum_j T_{ij} \quad \forall i \tag{2-2}$$

$$b_j = \sum_i T_{ij} \quad \forall j$$

$$\text{and} \quad T_{ij} \geq 0 \quad \forall (i,j) \tag{2-3}$$

When the transportation costs C_{ij} are not constant, the equilibrium conditions remain the same as in (2-1), but the term C_{ij} becomes $C_{ij}(T_{ij})$ and these conditions become (see FLorian and Los, 1982):

$$\begin{aligned} S_i(d_i) + C_{ij}(T_{ij}) - I_j(b_j) &= 0 \quad \text{if} \quad T_{ij} > 0 \quad \forall (i,j) \\ &\geq 0 \quad \text{if} \quad T_{ij} = 0 \quad \forall (i,j) \,. \end{aligned} \tag{2-4}$$

In the case of the GSPEP, the transportation cost between a pair (i,j), on a path k, say C_k, is the sum of the transportation costs of the arcs of the path k:

$$C_k = \sum_{a \in k} C_a \tag{2-5}$$

where the indices a refer to arcs of the network, and C_a to transportation costs on arcs of the network. The conditions (2-4) then become (see Florian and Los, 1982), for the GSPEP:

$$\begin{aligned} S_i(d_i) + \sum_{a \in k} C_a(v) - I_j(b_j) &= 0 \quad \text{if} \quad h_k > 0 \quad \begin{matrix} \forall (i,j) \\ \forall k \in K_{ij} \end{matrix} \\ &\geq 0 \quad \text{if} \quad h_k = 0 \end{aligned} \tag{2-6}$$

where:

K_{ij} is the set of paths connecting i to j,

h_k is the flow transported from i to j on path k,

v_a is the flow on arc a, and v the vector of flows v_a.

It is helpful to note that:

$$v_a = \sum_{i,j} \sum_{k \in K_{ij}} \delta_{ak} h_k \qquad (2\text{-}7\text{-}a)$$

where $\delta_{ak} = \begin{cases} 1 & \text{if arc a belongs to path k (between a given pair (i,j)),} \\ 0 & \text{otherwise.} \end{cases}$

The conservation of flow equations (2-2) for a general network becomes:

$$b_\ell + \sum_{a \in A_\ell} v_a - \sum_{b \in B_\ell} v_b - d_\ell = 0 \qquad \forall \ell \qquad (2\text{-}7\text{-}b)$$

and v must be positive: $v \geq 0$. (2-7-c)

In (2-7-b), A_ℓ represents the set of arcs which have node ℓ as tail node, and B_ℓ the set of arcs which have node ℓ as head node.

Before going on to the formulation of the GSPEP, we would like to introduce a new notation which has two main advantages. First it permits to simplify the notation used until now, and second, it allows to see the GSPEP as a Network Equilibrium Problem with one commodity (see also Dafermos, 1983).

$$\text{Let } S_i = C_{mi} \text{ and } d_i = v_{mi} \qquad \forall i$$
$$\text{and } -I_j = C_{jm} \text{ and } b_j = v_{jm} \qquad \forall j$$

Using this new notation, (2-6) becomes:

$$C_{mi}(v_{mi}) + \sum_{a \in k} C_a(v) - C_{jm}(v_{jm}) = 0 \quad \text{if} \quad h_k > 0 \quad \forall (i,j)\ \forall k \in K_{ij} \qquad (2\text{-}8)$$
$$\geq 0 \quad \text{if} \quad h_k = 0$$

One can understand that we (ficticiously) add to the original network a new node m and new arcs (m,i) and (j,m) for every i, and every j. Conditions (2-8) can then be rewritten:

$$\sum_{a \in k} C_a(v) = 0 \quad \text{if} \quad h_k > 0 \quad \forall k \in K$$
$$> 0 \quad \text{if} \quad h_k = 0 \quad \forall k \in K \qquad (2\text{-}9)$$

where the arcs a now belong to the augmented set A^+, the original set of arcs a, augmented by the arcs (m,i) and (j,m), $i \in O$, $j \in D$, and the paths k must now be seen as paths from m to m, passing through O/D pairs (i,j). Figure 2-1 illustrates the passage from the original network to the augmented network, for a brief example.

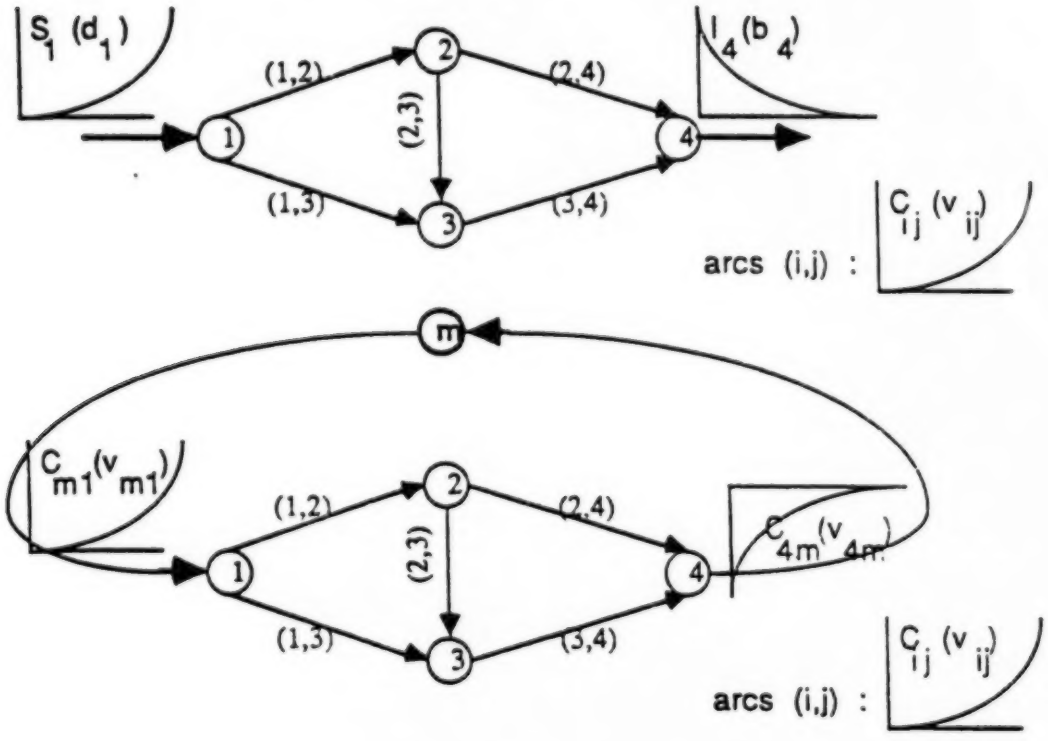

Figure 2-1: Passage from the original to the augmented network

The d_i's and b_j's are integrated in the v_a's, under this view, the constraints 2-7-b and 2-7-a become:

$$\sum_{a \in A_\ell} v_a - \sum_{b \in B_\ell} v_b = 0 \qquad \forall \ell \tag{2-10}$$

$$v \geq 0 \tag{2-11}$$

where the sets A_ℓ and B_ℓ are as defined before but seen in the augmented network.

Using a matrix notation, (2-10) may be written as:

$$Av=b \tag{2-12}$$

where A is the node-arc incidence matrix, and, in this particular case, b is equal to zero.

Conditions (2-10) are known to be equilibrium conditions of the Network Equilibrium Problem (NEP) on the augmented network, using one origin-destination pair (m,m). In a previous paper (Drissi and Florian, 1987) we show that this formulation of the GSPEP may be very practical for solving this problem in terms of path flow variables.

In the following, we use these simplified equilibrium conditions to formulate the problems (GSPEP and NEP) as variational inequalities. Note that this formulation is not new (see, for instance, Nguyen and Dupuis, 1984; or Friesz and Tobin, 1983):

We first note that (2-10) can be written:

$$C_k(h)\, h_k = 0,\ C_k \geq 0,\ h_k \geq 0 \qquad \forall k \in K \tag{2-13}$$

where $$C_k(h) = \sum_{a \in k} C_a(v) \tag{2-14}$$

and v as in (2-7).

Using the equivalence between complementary problems on a positive orthant and variational inequalities (Pang and Chan, 1982, Karamardian, 1971), (2-13) is equivalent to finding $\bar{h} \geq 0$ such that:

$$C_k(\bar{h})(h_k - \bar{h}_k) \geq 0 \qquad \forall h_k \geq 0 \qquad \forall k \in K \tag{2-15}$$

$$\text{or} \quad c(\bar{h})(h - \bar{h}) \geq 0 \qquad \forall h \geq 0 \tag{2-16}$$

where c is the vector of costs C_k, and h the vector of flows h_k. (2-16) is the path flow variable formulation, and in terms of arc flow variables, for every arc a, we have from (2-14) and (2-15) that:

$$\sum_{a \in k} C_a(\bar{v})(h_k - \bar{h}_k) \geq 0 \qquad \forall h_k \geq 0 \qquad \forall k \in K \tag{2-17}$$

and since $h \geq 0$ and $C_a(\bar{v}) \geq 0 \quad \forall a$, we obtain, for every $a \in A^+$, from (2-17) that:

$$C_a(\bar{v})(h_k - \bar{h}_k) \geq 0 \qquad \forall h_k \geq 0 \qquad \forall k \in K \quad \forall a \in A^+ \tag{2-18}$$

if we sum (2-18) on all the paths sharing arc a,

$$\sum_{k \in K} \delta_{ak} C_a(\bar{v})(h_k - \bar{h}_k) \geq 0 \qquad \forall h_k \geq 0 \qquad \forall k \in K \tag{2-19}$$

then, using (2-7):

$$C_a(\bar{v}) \cdot (v_a - \bar{v}_a) \geq 0 \qquad \forall h_k \geq 0 \qquad \forall a \in A \qquad \forall k \in K \ . \tag{2-20}$$

Note that $h_k \geq 0 \quad \forall k$ is equivalent to $v_a \geq 0 \quad \forall a \in A^+$. If we let C be the vector of costs C_a, $a \in A^+$, (2-20) becomes:

$$C(\bar{v})^T(v - \bar{v}) \geq 0 \qquad \forall v \geq 0 \ . \tag{2-21}$$

We recall that from (2-7), v must satisfy the flow conservation conditions, and it follows that the variational inequality to be solved to obtain an equilibrium $\bar{v}$, is:

$$\text{find} \quad \bar{v} \in \Phi \ : \ C(\bar{v})^T(v - \bar{v}) \geq 0 \qquad \forall v \in \Phi \tag{2-22}$$

where $\Phi = \{v \in R^n : Av=b,\ v \geq 0\}$, the set of flows satisfying constraints (2-10) (or (2-12)) and (2-11); $\bar{v}$ is said to be an equilibrium solution or an optimal solution. In the remainder of this paper, we denote (2-22) by $VI(C,\Phi)$.

In the next section, we consider a Jacobi-Newton type algorithm to solve this kind of problem.

3. LINEARIZED-JACOBI-NEWTON FOR VI(C,Φ)

We recall that the variational inequality VI(C,Φ) consists in finding $v^* \in \Phi$ such that $C(v^*)^T(v-v^*) \geq 0$ for any $v \in \Phi$, where $\Phi=\{v \in R^n \mid Av=b,\ v \geq 0\}$.

We suppose A:mxn ($m<n$), a full rank matrix, b:mx1, and C is continuous and differentiable on Φ.

In this section, we recall some well-known results on variational inequalities and linearized-Jacobi type algorithms. Then, we explore the combination of those results with a Newton method, and present an efficient linearized-Jacobi-Newton (LJN) algorithm to solve the problems of the form VI(C,Φ).

Theorem 3.1: Let $x^* \in \Phi$ be the optimal solution to VI(C,Φ).
Then x^* is optimal solution to: $\min C(x^*)^T x$ s.t. $x \in \Phi$.

Proof: VI(C,Φ): $x^* \in \Phi$ and $C(x^*)^T(x-x^*) \geq 0$ for any $x \in \Phi$

: $x^* \in \Phi$ and $C(x^*)^T x \geq C(x^*)^T x^*$ for any $x \in \Phi$

: $x^* = \operatorname{Arg} \min_{x \in \Phi} C(x^*)^T x$. ∎

Let $C^k(x) = C(x^k) + \Lambda(x^k)(x-x^k)$ (3-1)

be an approximation to C(x) at $x^k \in \Phi$, and consider the algorithm:

Algorithm 3.1: (General iteration): Given $x^k \in \Phi$, then let x^{k+1} be the optimal solution to VI(C^k,Φ).

In the particular case where $\Lambda(x^k) = \nabla C(x^k)$ is the Jacobian of C, evaluated at x^k, algorithm 3.1 corresponds to the Newton method to solve VI(C,Φ). Then the algorithm converges quadratically and, under mild conditions, it is proved by Pang and Chan (1982) that the algorithm has global convergence. But since ∇C is not necessarily symmetric, solving the subproblem VI(C^k,Φ) is not easier than solving the problem VI(C,Φ).

According to the symmetry principle (Ortega and Reinbolt, 1970, p. 95), if Λ is symmetric, then one can find a function $f:R^n \to R$ such that $\nabla^2 f = \Lambda$. This may be very helpful in the sense that one can replace the subproblem $VI(C^k,\Phi)$ by a mathematical program as we shall see later. An efficient mathematical programming method may be used at this moment to solve the subproblem.

In the remainder of this paper, we consider the case where $\Lambda(x^k)=D_k$, the diagonal part of $\nabla C(x^k)$. Algorithm 3.1, with $\Lambda(x^k)=D_k$, is the linearized-Jacobi algorithm, and Pang and Chan (1982) proved that this algorithm converges (locally), under the conditions that Φ is closed, convex, nonempty, and that C is continuous and differentiable, and that $\nabla C(x^k)$ is row and column diagonally dominant (strictly under row or column) and has positive (diagonal) elements.

In the following, we prove that the subproblem $VI(C^k,\Phi)$, using $\Lambda(x^k)=D_k$, may be formulated as a quadratic problem. Then we use a Newton-Active-Constraints (Drissi and Florian, 1987b) type algorithm, to solve $VI(C,\Phi)$, a Linearized-Jacobi-Newton (LJN) algorithm.

Theorem 3.2: Let
$$C^k(x) = C(x^k) + D_k(x-x^k) \tag{3-2}$$
where D_k is the diagonal part of $\nabla C(x^k)$ or any symmetric positive definite matrix, and $\Phi = \{x \in R^n: Ax=b, x \geq 0\}$. Then the optimal solution to:
$$(P_k): \min_{x\in\Phi} (C(x^k) - D_k x^k)^T x + \tfrac{1}{2} x^T D_k x \tag{3-3}$$
is also optimal solution to $VI(C^k,\Phi)$. ∎

Proof: Let $\bar{x}$ be the solution to P_k. Then $\bar{x}$ satisfies the Kuhn-Tucker conditions:
$$C(x^k) - D_k x^k + D_k \bar{x} + A^T \bar{\lambda} \geq 0 \tag{3-4}$$

$$\bar{x}^T(C(x^k) - D_k x^k + D_k \bar{x} + A^T \bar{\lambda}) = 0 \qquad (3\text{-}5)$$

$$A\bar{x}=b \qquad (3\text{-}6)$$

$$\bar{x}>0 \qquad (3\text{-}7)$$

Now let $\tilde{x}$ be the solution to VI(C^k,Φ). By using theorem 3.1, $\tilde{x}$ is solution to:

$$\min C^k(\tilde{x})^T x \quad \text{s.t. } x\varepsilon\Phi \qquad (3\text{-}8)$$

and then satisfies to Kuhn-Tucker conditions relative to (3-8):

$$C^k(\tilde{x}) + A^T \tilde{\lambda} \geq 0 \qquad (3\text{-}9)$$

$$\tilde{x}^T(C^k(\tilde{x}) + A^T \tilde{\lambda}) = 0 \qquad (3\text{-}10)$$

$$A\tilde{x}=b \qquad (3\text{-}11)$$

$$\tilde{x}>0 \quad . \qquad (3\text{-}12)$$

By relation (3-2), $C^k(\tilde{x}) = C(x^k) + D_k(\tilde{x}-x^k)$ that we replace in (3-9) to 3(10) to obtain:

$$C(x^k) + D_k(\tilde{x}-x^k) + A^T \tilde{\lambda} \geq 0 \qquad (3\text{-}13)$$

$$\tilde{x}^T(C(x^k) + D_k(\tilde{x} -x^k) + A^T \tilde{\lambda}) = 0 \qquad (3\text{-}14)$$

$$A\tilde{x}=b \qquad (3\text{-}15)$$

$$\tilde{x}>0 \quad . \qquad (3\text{-}16)$$

It is clear that by taking $(\tilde{x}, \tilde{\lambda})=(\bar{x}, \bar{\lambda})$ the solution to (P_k), (3-13) to (3-16) are satisfied because $(\bar{x}, \bar{\lambda})$ satisfy (3-4) to (3-17). ∎

Using theorem 3.2, we can now rewrite algorithm 3.1 as:

<u>Algorithm 3.2</u>: (general interation): given $x^k\varepsilon\Phi$, let $x^{k+1}=\bar{y}$ be the optimal solution to:

$$(P_k): \begin{cases} \min\ (C(x^k) - D_k x^k)^T y + \tfrac{1}{2} y^T D_k y & (3\text{-}17) \\ \text{s.a.}\quad Ay = b & (3\text{-}18) \\ y \geq 0 & (3\text{-}19) \end{cases}$$

One of the most efficient methods to solve a quadratic program like P_k is the active-set method that we combine with the Newton algorithm. Using an initial feasible point $y^0 = x^k$ and an initial active set w^0, the method consists in moving along the Newton direction, restricted to current active set, until it finds the optimal active set, and then, the Newton direction gives the optimal solution $\bar{y}$ to (P_k). As formulated in (Luenberger, 1984, p. 626), this algorithm to solve (P_k) can be stated as:

Algorithm 3.3: 0 - Initialize $y^0 = x^k \in \Phi$ and w^0 = the set of initial active constraints (i.e. equality constraints plus some constraints $y_i = 0$); $\ell = 0$.

1 - Solve the quadratic program subject to the constraints in w^ℓ (by a Newton iteration). If the Newton direction $p^\ell = y^{\ell+1} - y^\ell = 0$, then go to 3.

2 - Let $y^{\ell+1} = y^\ell + \alpha^\ell p^\ell$ where $\alpha^\ell = \min(1, \bar{\alpha}^\ell)$ where $\bar{\alpha}^\ell = \max\{\alpha > 0 \mid y^\ell + \alpha p^\ell$ feasible relatively to nonnegativity constraints$\}$

If $\alpha^\ell < 1$, then add to w^ℓ the new active constraints; $\ell = \ell + 1$ and go to 1.

3 - Let λ be the multiplier vector associated to the constraints in w^ℓ, and $\lambda_q = \min\{\lambda_i \mid i \in w^\ell \cap \{$inequality constraints$\}\}$

If $\lambda_q > 0$ then y^ℓ is optimal; stop. Otherwise drop q from w^ℓ; $\ell = \ell + 1$ and go to 1.

In the remainder of this subsection, we propose a Newton algorithm combined with an active-set strategy to solve P_k. First, we would like to mention that (P_k) is a quadratic program, and secondly, if one knew the optimal active set of (P_k), only one Newton iteration (restricted to the optimal active set) would give the optimal solution to (P_k). On the other hand, the previous active set method for quadratic programming (Luenberger, (1984), p. 626) consists in choosing an initial active set w^o, and in modifying w^o until the optimal set w^* is found. In Drissi (1987), we propose a choice of w^o based on the value of $C(x^k)$ and on the multiplier values λ^k. Numerical results, as it will be shown later, show that w^o, chosen that way, is a good approximation of w^*.

We define a set I by:

$$I = \{ i \mid y_i^o = x_i^k = 0 \quad \text{and} \quad C_i(x^k) + a_i^T \lambda^\ell > 0 \} .$$

For simplifying the notation we use the same notation I for the set of indices as defined above and the restriction of matrix A to the columns corresponding to elements in I. This will be true also for A and J where $J = A \setminus I$ is the set of indices in A not in I. Then, the constraints used in w^o are:

$$Ay = b \tag{3-20}$$

$$y_i = 0, \quad i \in I \tag{3-21}$$

The main interation of algorithm 3.3 (step 1) will consist in solving programs having the form:

$$\min \quad (C(x^k) - D_k x^k)^T y + \tfrac{1}{2} y^T D_k y \tag{3-22}$$

$$\text{s.t.} \quad Ay = b \quad (\lambda^\ell) \tag{3-23}$$

$$-y_I = 0 \tag{3-24}$$

where y_I is the restriction of y to the elements in I. Let the matrix A=[J|I] be partitioned into elements in J and I. The solution $y^{\ell+1}$ to (3-22) to (3-24) is obtained by solving:

$$C(x^k) - D_k x^k + D_k y^{\ell+1} + \left[\begin{array}{c|c} J^T & 0 \\ \hline I^T & -Id \end{array}\right] \left[\begin{array}{c} \lambda^\ell \\ \hline \mu^\ell \end{array}\right] = 0 \qquad (3\text{-}25)$$

$$Jy_J^{\ell+1} + Iy_I^{\ell+1} = b \qquad (3\text{-}26)$$

$$y_I^{\ell+1} = 0 \qquad (3\text{-}27)$$

where Id is the identity matrix relative to elements in I. If we adopt the notation $p^\ell = y^{\ell+1} - y^\ell$, this system becomes:

$$D_k\, p^\ell + \left[\begin{array}{c|c} J^T & 0 \\ \hline I^T & -Id \end{array}\right] \cdot \left[\begin{array}{c} \lambda^\ell \\ \hline \mu^\ell \end{array}\right] = \underbrace{-(C(x^k) - D_k x^k + D_k y^\ell)}_{-\delta^\ell} \qquad (3\text{-}28)$$

$$Jp_J^\ell + Ip_I^\ell = 0 \qquad (3\text{-}29)$$

$$p_I^\ell = 0 \qquad (3\text{-}30)$$

Decomposing (3-28) along elements in I and J, and replacing (3-30) in the equations leads to:

$$D_J^k\, p_J^\ell + D_{IJ}^k\, p_I^\ell + J^T\lambda^\ell = -\delta_J^\ell \qquad (3\text{-}31)$$

$$D_{IJ}^k\, p_J^\ell + D_I^k\, p_I^\ell + I^T\lambda^\ell - \mu^\ell = -\delta_I^\ell \qquad (3\text{-}32)$$

$$Jp_J^\ell = 0 \qquad (3\text{-}33)$$

$$p_I^\ell = 0 \qquad (3\text{-}34)$$

We recall that D_k is diagonal so that this system is equivalent to

$$D_J^k\, p_J^\ell + J^T\lambda^\ell = -\delta_J^\ell \qquad (3\text{-}35)$$

$$Jp_J^\ell = 0 \qquad (3\text{-}36)$$

$$p_I^\ell = 0 \qquad (3\text{-}37)$$

$$\mu^\ell = \delta_I^\ell + I^T\lambda^\ell \qquad (3\text{-}38)$$

The whole system is then reduced to (3-35) to (3-36) and equations (3-37) and (3-38) are trivial to solve.

On the other hand, let's multiply (3-35) by $J(D_J^k)^{-1}$:

$$J(D_J^k)^{-1}\ D_J^k\ p_J^\ell + J(D_J^k)^{-1}\ J^T\lambda^\ell = -J(D_J^k)^{-1}\delta_J^\ell \qquad (3\text{-}39)$$

and since $Jp_J^\ell = 0$, then $J(D_J^k)^{-1}\ J^T\lambda^\ell = -J(D_J^k)^{-1}\ \delta_J^\ell$ (3-40)

$$\text{or} \quad \lambda^\ell = -(J(D_J^k)^{-1}\ J^T)^{-1}\ J(D_J^k)^{-1}\ \delta_J^\ell \qquad (3\text{-}41)$$

so that, from (3-35):

$$p_J^\ell = -(D_J^k)^{-1}(J^T\lambda^\ell + \delta_J^\ell) \qquad (3\text{-}42)$$

The system (3-28) to (3-30) is then reduced to solving one "little" system (3-40) and three "trivial" equations (3-42) (because D_J^k is diagonal), (3-37) and (3-38).

To "visualize" the whole algorithm (LJN), we present it in a block diagram form in figure 3.1.

Important Comments:

a) The first comment concerns capacity constraints ($v<u$). LJN can handle this type of constraint in the same way that nonnegativity constraints are handled. Yet, two modifications must be brought to LJN:

- Choice of I: the initial active set must contain, like before the set E of equality constraints, the set I_o of the constraints $v_i=0$ (or $p_i=0$) where i is such that $v_i^k=0$ and $C_i(v^k) + a_i^T\lambda^k > 0$ to which we add the set I_u of constraints $v_i = u_i$ (or $p_i=0$) where i is such that $v_i^k=u_i$ and $C_i(v^k) + a_i^T\lambda^k < 0$. $w^o = E \cup I_o \cup I_u = E \cup I$.
- Feasibility of the direction p^k: the test must be done on the constraints $v>0$ (i.e. $v_i^k=0$ and $p_i^k<0$) and also on $v<u$ (i.e. $v_i^k=u$ and $p_i^k>0$).

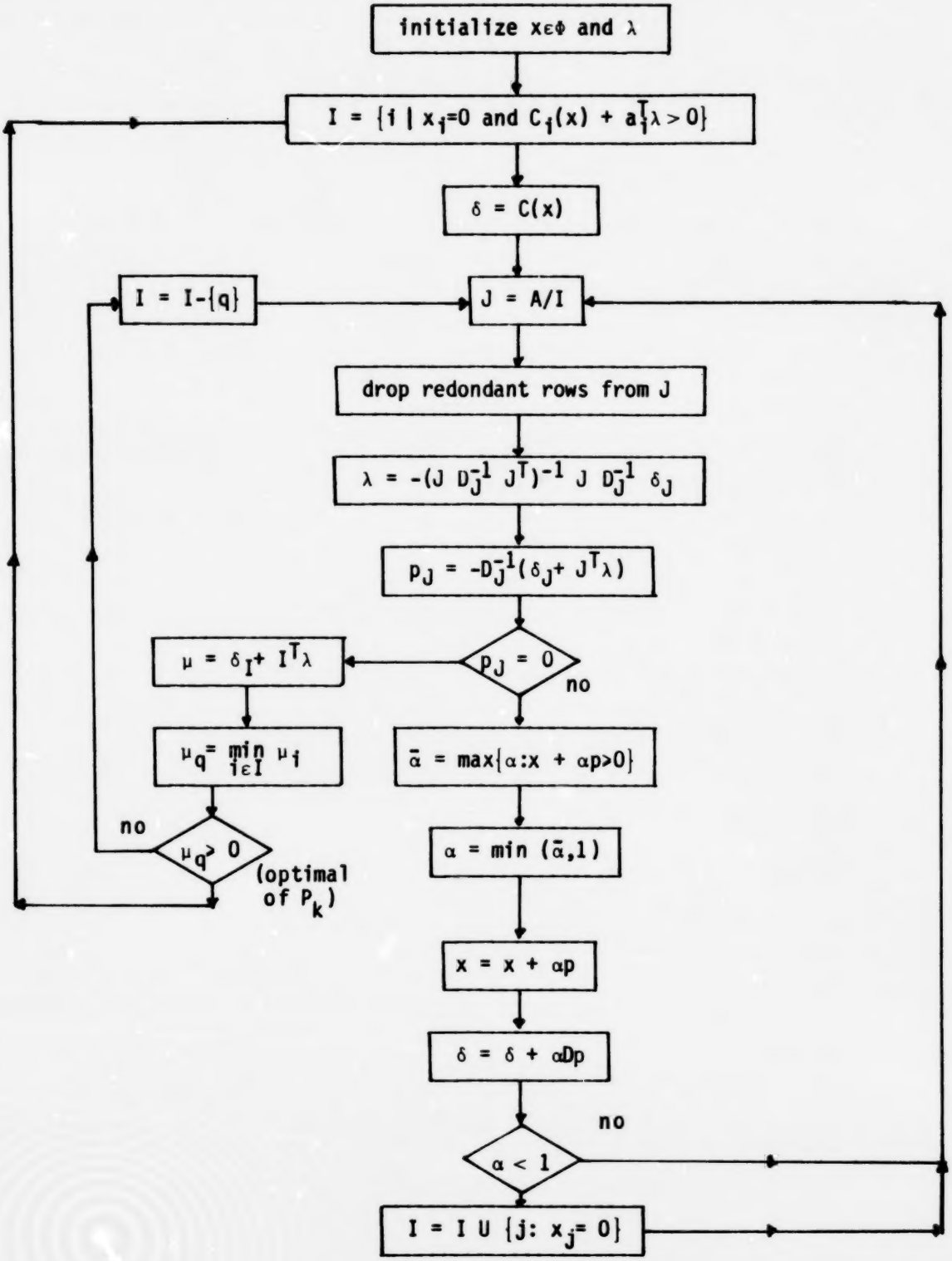
initialize xεΦ and λ
I = {i | x_i=0 and C_i(x) + a_i^T λ > 0}
δ = C(x)
I = I-{q}
J = A/I
drop redondant rows from J
λ = -(J D_J^-1 J^T)^-1 J D_J^-1 δ_J
p_J = -D_J^-1(δ_J + J^T λ)
μ = δ_I + I^T λ
p_J = 0
no
μ_q = min_{i∈I} μ_i
ᾱ = max{α:x + αp>0}
no
μ_q > 0
(optimal of P_k)
α = min (ᾱ,1)
x = x + αp
δ = δ + αDp
no
α < 1
I = I U {j: x_j = 0}

Figure 3.1: Block diagram of algorithm LJN

b) As it has been shown in the last section, nonnegativity constraints may be used to reduce the system to be solved at each iteration. Capacity constraints also do so, because they are treated in the same way as nonnegativity constraints.

c) We would like to note that in (LJN), the Jacobian ∇C is approximated at each interation by its diagonal part. In the important case of separable problems, where $C_i(v) = C_i(v_i)$, i=1..., the Jacobian ∇C is equal to its diagonal part. Then, in this case, LJN is a pure Newton method to solve the problem. We also note that this problem is equivalent to a mathematical program (P):

$$(P) : \min \sum_i \int_0^{v_i} C_i(x)dx$$

$$\text{s.t.} \quad Av=b$$

$$v \geq 0$$

$$(v \leq u) \quad .$$

4. ADAPTATION TO THE GSPEP IN TERMS OF ARC FLOW VARIABLES

In a preceding section, we have seen that the GSPEP may be stated as a NEP:

$$VI(C,\Phi) : \text{find } v^* \in \Phi \text{ such that } C(v^*)^T(v-v^*) \geq 0 \text{ for any } v \in \Phi$$

where $\Phi = \{v \geq 0 \mid \sum_{a \in A_\ell} v_a - \sum_{b \in B_\ell} v_b = r_\ell \text{ for any } \ell\}$.

In the particular case of the GSPEP, the second number r is equal to zero, r=0. The problem $VI(C,\Phi)$ is the kind of problem for which LJN has been designed in the last section. Then, to solve the GSPEP (or NEP), by LJN, the steps that need a particular care are the following:

- Initialization: for the GSPEP, v=0 is a feasible solution (because r=0) and thus it may be used as an initial solution. For the NEP, an initial solution may be obtained by computing the shortest path between every origin-destination pair (i,j), and by affecting on each of these shortest paths the whole quantity to be transported from the origin to the destination. If $v > 0$, then the initial value of λ has no importance; if not, λ may be initialized by using equation 3-41.

- Computation of $C + A^T\lambda$: consider a column corresponding to the arc a=(i,j), then the only nonnull elements of that column are those corresponding to the nodes i and j, with the values -1 for i and 1 for j. Then $C + A^T\lambda = [C_a(v) - \lambda_i + \lambda_j]_{a \in A^+}$. Note that at the current iteration, λ_ℓ is set equal to zero for the unused nodes ℓ that are defined as follows:

- Unused rows of A (unused nodes): relatively to the computation of $C+A^T\lambda$ and to the active set strategy, we have seen in LJN how to restrict computations to a subset of variables (subset I of arcs). When we drop arcs $a \in I=A/J$, some nodes may be disconnected from the network. Flow conservation conditions for these nodes become unnecessary; these nodes may be obtained by the following algorithm:

0 - label arcs $a \in J$ by $p(a)=1$
and arcs $a \notin J$ by $p(a)=0$
Label all nodes k by $PN(k)=0$ and let $k=m$

1 - Let $PN(k)=2$

2 - For every arc $a=(i,k)$ or (k,i) such that $p(a)=1$ do:
if $PN(i) \neq 2$, then $PN(i)=1$

3 - If there is i such that $PN(i)=1$, let $k=i$ and go to 1.
Otherwise stop.

At the end of this procedure, all the nodes i connex to the supernode m are labeled with $PN(i)=1$ or 2. The others are not used.

- Solution of the system: $(JD_J^{-1}J^T)\lambda = -JD_J^{-1}\delta_J$

i) second term: the element of $-JD_J^{-1}\delta_J$ corresponding to the node ℓ is computed as follows:

$$(-JD_J^{-1}\delta_J)_\ell = -\sum_{a \in A_\ell \cap J} \delta_a / C'_a(v) + \sum_{b \in B_\ell \cap J} \delta_b / C'_b(v) \quad \forall \ell$$

ii) solution of the system: two methods may be used. The first may be an exact method like the Gauss method (Conte and De Boor, 1980), but this method needs the knowledge of the matrix $JD_J^{-1}J^T$, and this may be quite demanding (in storage) for large networks. In this case, an iterative method like the conjugate gradient method (Berksekas, 1982) is useful because this method does not need explicitly the matrix $JD_J^{-1}J^T$, but product of the form $JD_J^{-1}J^T y$ where y is a known vector.

- Computation of $JD_J^{-1}J^T$ (for the Gauss method): Details of implantation of the Gauss method can be found in literature (see Conte and De Boor, 1980, for instance), but since D is diagonal and the node-arc matrix has a particular structure, $JD_J^{-1}J^T$ may be computed efficiently by:

o - M←0

i - For every arc $a=(i,j)\in J$, do:

$M_{ii} \leftarrow M_{ii} + 1/C'_a$

$M_{jj} \leftarrow M_{jj} + 1/C'_a$

$M_{ij} \leftarrow M_{ij} - 1/C'_a$

$M_{ji} \leftarrow M_{ji} - 1/C'_a$

where C'_a, $a\in J$, is the partial derivative of C_a, relatively to v_a: $C'_a = \partial C_a(v)/\partial v_a$.

- The conjugate gradient algorithm also can be found in literature (see Bertsekas, 1982, for instance). Here, we need not know explicitly the matrix $JD_J^{-1}J^T$ but products of the form $p=JD_J^{-1}J^Ty$ that can be efficiently computed by:

o - p←0

i - For every $a=(i,j)\in J$, do:

$p_i \leftarrow p_i + (y_i - y_j)/C'_a$

$p_j \leftarrow p_j - (y_i - y_j)/C'_a$

The other steps of LJN do not need comments on their adaptation to GSPEP or NEP.

The next section presents numerical results relative to LJN.

5. NUMERICAL RESULTS

We implemented the LJN algorithm on the CYBER 835 of the Université de Montréal, via a Fortran code. Most of the networks studied have been randomly generated, but before presenting them, we propose to come back on the theoretical context that suggested a way to generate problems.

Recall that in LJN, the D matrix is the diagonal part of the Jacobian ∇C. In the particular case of a separable problem (i.e. $C_a(v)=C_a(v)$), the Jacobian matrix ∇C is diagonal, and in this case LJN is exactly the Newton method with quadratic convergence. On the other hand, the Newton method with the strategy of Active Constraints (NAC) has been shown (Drissi, Drissi and Florian, 1987) to be of the best algorithms for the separable GSPEP. Then, the approach used to test the efficiency of LJN is to consider first a separable problem for which LJN represents practically the state of the art, and then to introduce "asymmetries" and see their effect on LJN.

One parameter to evaluate asymmetry is the coefficient δ which measures the number of unequal symmetrical elements in the Jacobian matrix:

$$\delta = \frac{\text{number of unequal symmetrical element (out of diagonal)}}{\text{number of out of diagonal elements}} \times 100 \qquad (5\text{-}1)$$

Four networks are used:

1 - Nguyen and Dupuis' Network (1984): this is a network equilibrium problem with 13 nodes, 19 arcs, and 2x2 O/D pairs.
$C=10^{-3}Ax+b$ is linear, so the Jacobian matrix $\nabla C=10^{-3}A$ is constant. For the separable problem, we restrict A to its diagonal part.

2 - The second problem uses the same network as the first one, but to transform it into a GSPEP, we replace the fixed supply and demand by supply functions at origins and demand functions at destinations:

origins: node 1: $S_1(d_1) = d_1^2/400$ (5-2)

node 4: $S_4(d_4) = d_4^2/500$ (5-3)

destinations: node 2: $I_2(b_2) = 10^5/(1+b_2)$ (5-4)

node 3: $I_3(b_3) = 2.10^5/(1+b_3)$ (5-5)

3 - Random problem: A network is randomly generated with 10 nodes, 32 arcs, 5x5 O/D pairs.

The functions used are:

at origins: $S_i(d_i) = a_i + b_i d_i + c_i d_i^2$ (5-6)

where a_i and $b_i \in [0,1]$ and $c_i \in [0.01,0.1]$ are randomly generated.

at destinations: $I_j(b_j) = a_j/(1+b_j)$ (5-7)

where $a_i \in [20,50]$ are randomly generated.

on the arcs: $C_a(v_a) = k_a + k'_a v_a + \sum_{i=1}^{3} (k_{a_i} + k'_{a_i} v_{a_i})$ (5-8)

where $k_a \in [0,1]$ and $k'_a \in [1,1.5]$ and k_{a_i} and $k'_{a_i} \in [0.1,0.3)$ are randomly generated. The sum $\sum_{i=1}^{3}$ means that the cost on an arc a depends on three other arcs a_i, i=1,...,3, randomly generated. For the symmetrical case, we fix k_{a_i} and k'_{a_i} at zero.

The coefficient δ is $\dfrac{32 \times 3 \times 100}{(32+5+5)^2-(32+5+5)} \approx 10\%$

4 - The last network used is described in Tobin and Friezs (1983), and has 5 nodes, 12 arcs, 5x5 O/D pairs.

The functions used on this network are:

at origins: $$S_i(d_i,v)\, E_i + B_i d_i^{n_i} + \sum_{\substack{j=1 \\ j\neq i}}^{N} k_j v_j \qquad (5\text{-}9)$$

where E_i, B_i and n_i are in Tobin and Friezs (1983).

at destinations: $$I_j(b_j,v) = A_j\, e^{-T_j b_j} - \sum_{\substack{i=1 \\ i\neq j}}^{N} k_i v_i \qquad (5\text{-}10)$$

where A_j and T_j are in Tobin and Friezs (1983).

on the arcs: $$C_a(v) = n_a + R_a v_a + \sum_{\substack{i=1 \\ a\neq i}}^{N} k_a v_a \qquad (5\text{-}11)$$

where n_a and R_a are in Tobin and Friezs (1983).

In the sums, the value of the coefficient N has an effect on the parameter δ. With N=22, δ=100%, with N=11, δ=30%, and with N=i in (5-9), N=j in (5-10), N=a in (5-11), δ=80%. At the same time, we can change the values of the coefficients k in the sums, having an effect on the magnitude of the asymmetry between symmetrical elements. We use three cases where $k\in[0.001,0.002]$, $k\in[0.001,0.011]$ and $k\in[0.001,0.101]$. The values of k may seem small but this is to keep diagonal dominance of the Jacobian.

The numerical results are presented in Table 1. These results are obtained with the following termination criterion (CT):

(CT): stop when $$C_i(v) + a_i^T\lambda > 0 \quad \text{if} \quad v_i < E_1 \qquad (5\text{-}12)$$

or $$|C_j(v) + a_j^T\lambda| < E_2 \quad \text{if} \quad v_j > 0 \qquad (5\text{-}13)$$

The tolerances E_1 and E_2 are $E_1 = E_2 = 10^{-4}$, but because of the good rate of convergence, the maximal value in the reduced gradient (in 5-13) is of the order of 10^{-8} to 10^{-12} at the termination.

In all problems, we note that the nonsymmetrical case (with $\delta \neq 0$) is solved requiring approximately the same CPU time and number of iterations that the symmetrical case (with $\delta = 0$).

The weak effect of asymmetry on LJN makes that LJN is an efficient algorithm as well for separable problems as for asymmetrical problems (with diagonal dominant Jacobian ∇C). Since, on the other hand, it is known (Drissi and Florian, 1987) that this kind of algorithm is one of the most efficient algorithm for separable problems, we conclude here from this and the discussion above, that LJN is one of the most efficient algorithms for solving variational inequalities (with diagonal dominant Jacobian ∇C).

Table 1 : LJN Numerical Results

Problem	type	N	A	O	D	NV	# iter.	CPU time (sec.)	δ(%)	N	# k
1.0	NE	13	19	2	2	19	2	0.45	0		
							8	0.75	23		
2.0	SE	13	19	2	2	23	11	1.11	0		
							13	1.38	16		
3.0	SE	10	32	5	5	42	6	1.07	0		
							8	1.03	10		
4.0	SE	5	12	5	5	22	6	0.48	0		0.000
4.1	SE	5	12	5	5	22	6	0.46	30	11	0.002
4.2	SE	5	12	5	5	22	6	0.47	30	11	0.011
4.3	SE	5	12	5	5	22	8	0.47	30	11	0.101
4.4	SE	5	12	5	5	22	7	0.41	80	a	0.002
4.5	SE	5	12	5	5	22	7	0.42	80	a	0.011
4.6	SE	5	12	5	5	22	6	0.43	80	a	0.101
4.7	SE	5	12	5	5	22	8	0.45	100	22	0.002
4.8	SE	5	12	5	5	22	6	0.40	100	22	0.011
4.9	SE	5	12	5	5	22	10	0.47	100	22	0.101

Type: NE = Network Equilibrium
SE = Spatial Equilibrium

where N : number of nodes;
A : number of arcs;
O : number of origins;
D : number of destinations;
NV : number of variables;
iter. : general iterations;
CPU time : (sec.);

δ : δ + diagonal, $\delta = \dfrac{\text{number of out of diagonal unequal symmetrical elements}}{\text{number of out of diagonal elements}} \times 100$;

k : $0.002 \rightarrow k_i \in [0.001, 0.002]$, $0.011 \rightarrow k_i \in [0.001, 0.011]$, $0.101 \rightarrow k_i \in [0.001, 0.101]$.

6. CONCLUSION

In this paper, we consider monotone variational inequalities with linear equality constraints and nonnegativity and capacity constraints. We develop a Linearized-Jacobi-Newton method with an active constraint strategy to solve this kind of problem.

On the other hand, we consider the General Spatial Price Equilibrium Problem (GSPEP) which consists in finding an equilibrium between the production costs at origins plus transportation costs on the arcs of the network, and the market price at destinations. We show that this problem can be seen by adding a node and some arcs to the original network, as a Network Equilibrium Problem (NEP). We consider the asymmetrical case where the cost on each arc may depend on the flow on other arcs. Then, the GSPEP and NEP may be written as a variational inequalities with linear equality constraints (flow conservation conditions) and nonnegativity and capacity constraints.

We use the proposed LJN algorithm to solve these problems, and the numerical results obtained from a Fortran code of LJN on the CYBER 835 of the Université de Montréal to demonstrate the efficiency of this algorithm.

ACKNOWLEDGEMENTS

This paper benefited from a careful reading by Michael Florian.

BIBLIOGRAPHY

BERTSEKAS, D. and GAFNI, E., "Projected Methods for Variational Inequalities with Application to the Traffic Equilibrium Problem", Mathematical Programming Study 17, 139-159 (1982).

CONTE, S.D. and DE BOOR, C., Elementary Numerical Analysis, an Algorithmic Approach, troisième édition, Chap. 4, Mc Graw Hill (1980).

DAFERMOS, S., "Traffic Equilibrium and Variational Inequalities", Transportation Science 14(1), 42-54 (1980).

DAFERMOS, S. and NAGURNEY, A., "Isomorphism between Spatial Price and Traffic Network Equilibrium Models", Lefschetz Center for Dynamical Systems, Brown University, Provident, RI LCDS #85-17 (1985).

DAFERMOS, S., "An Iterative Scheme for Variational Inequalities", Mathematical Programming 26, 40-47 (1983).

DRISSI-KAÏTOUNI, O., "Un algorithme pour le Problème d'Équilibre Spatial Général exprimé dans l'espace des flots sur les chemins", publication #448, Centre de recherche sur les transports, Université de Montréal (1986).

DRISSI-KAÏTOUNI, O., "Application de Méthodes de Newton au Problème d'Équilibre Spatial sur un Réseau Général", thèse de Ph.D., Département d'Informatique et de Recherche opérationnelle, Université de Montréal (1987).

DRISSI-KAÏTOUNI, O. and FLORIAN, M., "An Algorithm for the Spatial Price Equilibrium Problem in Term of Path Flow Variables", publication #517, Centre de recherche sur les transports, Université de Montréal (1987).

DRISSI-KAÏTOUNI, O. and FLORIAN, M., "An Active Constraints Newton Algorithm for Convex Cost Minimization with Application to the Spatial Price Equilibrium Problem", publication #537, Centre de recherche sur les transports, Université de Montréal (1987b).

FLORIAN, M., "On Some Generalizations of the Transportation Problem" Operational Research'81, P. Brans (ed.), IFORS, 607-620 (1981).

FLORIAN, M. and LOS, M., "A New Look at Static Spatial Price Equilibrium Models", Regional Science and Urban Economics 12, 579-597 (1982).

FRIESZ, T.L. and TOBIN, R.L., "Formulating and Solving the Spatial Price Equilibrium Problem with Transshipment in Terms of Arc Variables", Dept. of Civil and Urban Engineering, University of Pennsylvania, (1981).

FRIESZ, T.L., HARKER, P.T., and TOBIN, L., "Alternative Algorithms for the General Network Spatial Price Equilibrium Problem", Journal of Regional Science 24(4), 475-507 (1984).

HAMMOND, J., "Solving Asymmetric Variational Inequality Problems and Systems of Equations with Generalized Nonlinear Programming Algorithms", Ph.D. thesis, Massachusetts Institute of Technology (1984).

HITCHCOCK, F.L., "The Distribution of a Product from Several Sources to Numerous Localities", Journal of Mathematics and Physics 20, 224-230 (1941).

LUENBERGER, D.G., "Linear and Non Linear Programming", Chap. 6, 7, 11, Addison-Wesley, (1984).

MARCOTTE, P. and GUÉLAT, J., "A Modified Newton Method for Solving the Asymmetric Traffic Equilibrium Problem", publication #404, Centre de recherche sur les transports, Université de Montréal (1985).

MURTAGH, B.A. and SAUNDERS, M.A., "MINOS User's Guide", rapport technique 77-9, Systems Optimization Laboratory, Stanford University, Stanford, CA, (1977).

NAGURNEY, A., "Computational Comparisons of Spatial Price Equilibrium Methods", Journal of Regional Science 27(1), 55-76 (1987).

NGUYEN, S., "An Algorithm for the Traffic Assignment Problem", Transportation Science 8, 203-218 (1974).

NGUYEN, S. and DUPUIS, C., "An Efficient Method for Computing Traffic Equilibria in Networks with Asymmetric Transportation Costs", Transportation Science 18(2), 185-202 (1984).

ORTEGA, J.M. and RHEINBOLDT, W.C., "Iterative Solution of Non Linear Equations in Several Variables", Chap. 13, Academic Press, (1970).

PANG, J.S. and CHAN, "Iterative Methods for Variational and Complementary Problems", Graduate School of Industrial Administration, Carnegie-Mellon University (1982).

PANG, J.S. and CHAN, "Gauss-Seidel Methods for Variational Inequality Problems over Product Sets", Graduate School of Industrial Administration, Carnegie-Mellon University (1981).

SAMUELSON, P.A., "Spatial Price Equilibrium and Linear Programming", American Economics Review 42, 283-303 (1952).

SMITH, M.J., "The Existence, Uniqueness and Stability of Traffic Equilibria", Transportation Research 13B, 295-304 (1979).

TAKAYAMA, T. and JUDGE, G.C., "Spatial and Temporal Price and Allocation Models" North-Holland (1971).

WARDROP, J.G., "Some Theoretical Aspects of Road Traffic Research", Proceedings of the Institute of Civil Engineers, Part II 1, 325-378 (1952).

ZUBIETA, L., "Modèles d'Équilibre Spatial et méthodes de Résolution", thèse de Ph.D., département d'Informatique et de Recherche opérationnelle, Université de Montréal (1987).